Requirements and Design Specification Review

Command Support CSCI

Checkout and Launch Control Systems (CLCS)

84K00502

Approval:

Kirk Lougheed, Chief, System Engineering and Integration Division — Date

PREPARED BY: _____

_____

_____

_____

_____

_____

_____

_____

_____

REVISION HISTORY

REV	DESCRIPTION	DATE

LIST OF EFFECTIVE PAGES				
Dates of issue of change pages are:				
Page No.	**A or D***	**Issue or Change No.**	**CR No.**	**Effective Date****

Table of Contents

REQUIREMENTS AND DESIGN SPECIFICATION

REVIEW COMMAND SUPPORT CSCI

CHECKOUT AND LAUNCH CONTROL SYSTEMS (CLCS)

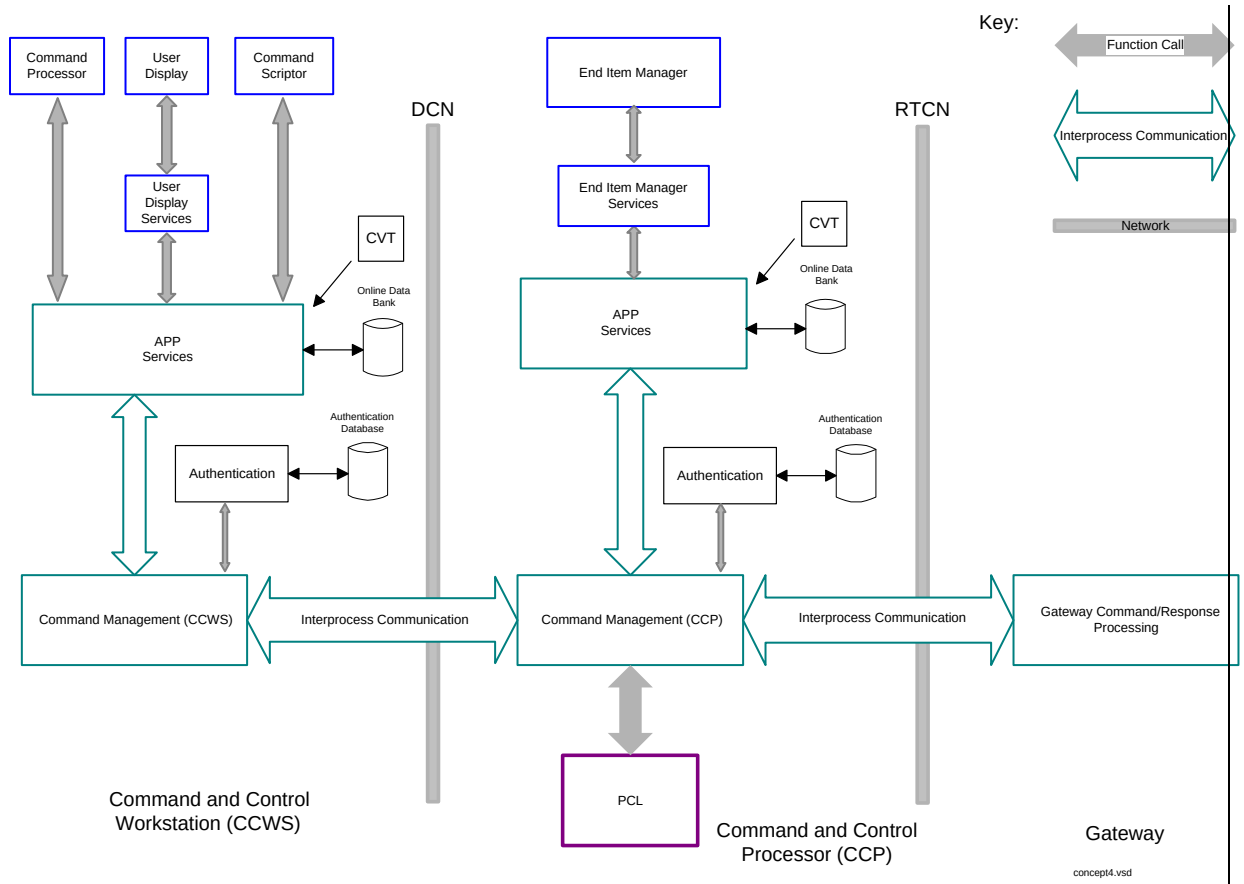
1. COMMAND SUPPORT CSCI

1.1 COMMAND SUPPORT INTRODUCTION

The Command Support CSCI provides users and user applications the ability to command the RTPS. The Command Support CSCI is detailed in the following sections: ~~consists of a command text entry user display, Command Processor, and a command authentication and routing function, Command Management.~~

Section 2 describes Command Processor, a command line user interface.
Section 3 describes Command Management, manages command transactions.
Section 4 describes Authentication Interface, the authentication API.
Section 5 describes Commanding Interface, the command API.
Section 6 describes the Command Scripter, a command text file interface.

1.1.1 Command Support Overview

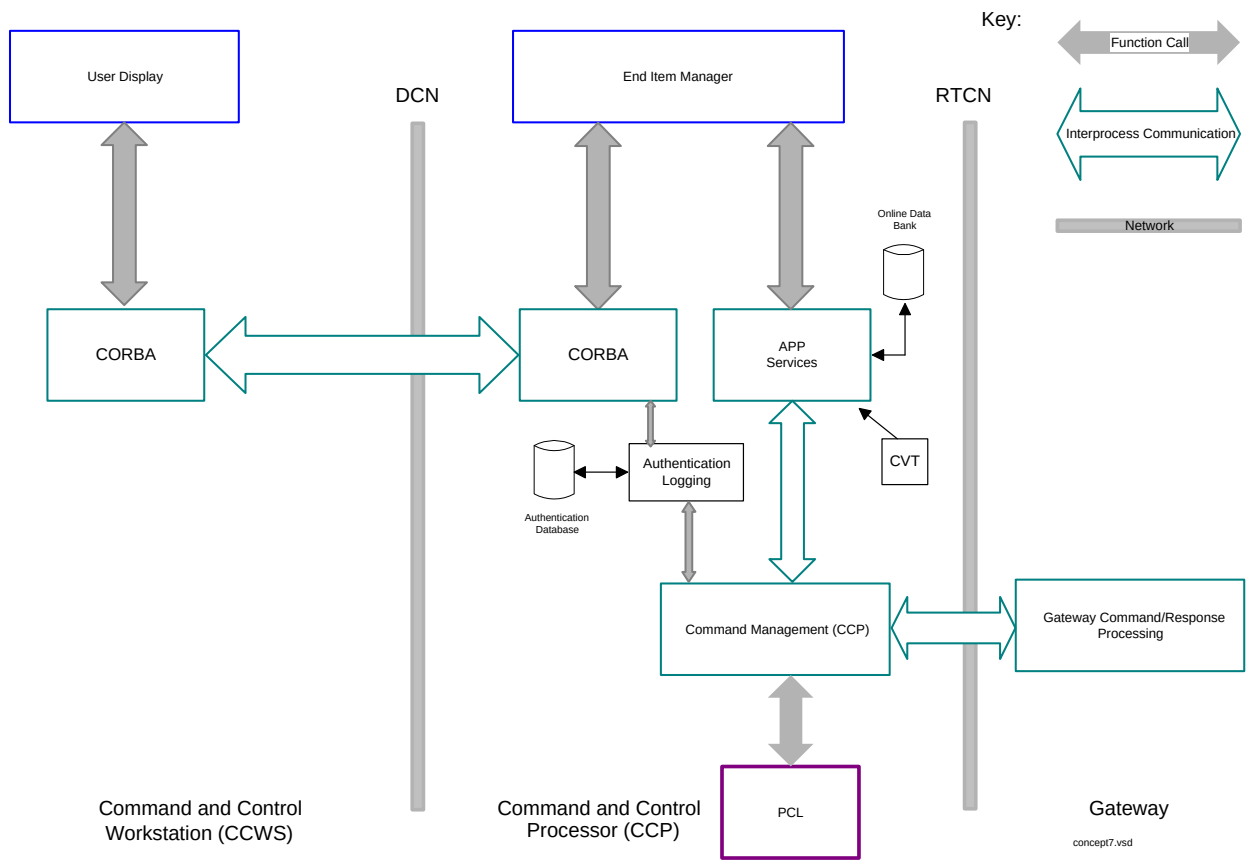


Command Support facilitates the user and user applications in commanding the RTPS in the Command and Control Workstation (CCWS) and Command and Control Processor (CCP). There are three flows of commanding. The first is through the Command Processor or

Command Scripter user interfaces on the CCWS. The second is through a GUI user application on the CCWS. The third is through a user application (EIM or TAS) executing in the CCP.

In the first command flow the ~~user interface,~~ Command Processor user interface~~,~~ accepts text ~~entry~~ input. The text is processed to form a command and then Applications Services is called to create and issue~~send~~ the command through the Command Interface. Commands are received by the CCWS resident Command Management CSC. Command Management validates the command authentication ~~of the user~~ and forwards the command to the CCP. A CCP resident Command Management CSC receives commands from CCWSs, performs authentication and FD PCL validation, and forwards ~~routes~~ the commands to the Gateways. Gateway Command Responses are sent back to the CCP Command Management and are then sent to the appropriate source.

In the second command flow, GUI applications command through "Hot Spots" on their screen. These applications interface with Application Services and the Commanding Interface to issue commands to the CCWS resident Command Management. From this point, command flow is similar to the Command Processor command flow.



The third command flow from a user application differs from the previous two flows in that commands originate from user applications executing on the CCP. The user applications

interface with Applications Services and Commanding Interface to create objects and issue commands. Commands are sent directly to the Command Management on the CCP where the command is authenticated for the application. Commands are then PCL checked before being routed to the appropriate Gateway. Gateway Command Responses are sent back to the CCP Command Management and are then sent to the appropriate source.

1.2 COMMAND SUPPORT SPECIFICATIONS

1.2.1 Command Support Common Ground Rules

N/A

~~**1.2.1.1 Command Support for Redstone will not include:**~~

~~A) Prerequisite Control Logic~~
~~A) Test Application Scripts~~
~~A) End Item Manager~~

1.2.2 Command Support Functional Requirements

1.2.2.1 The Command Support CSCI shall provide the RTPS capability to issue ~~two types of FD~~ commands listed in Appendix A, Command Syntax. ~~. The commands are ***Set*** for discrete FD's and the ***Apply*** for analog FD's.~~

1.2.3 Command Support Performance Requirements

See Command Support CSCs for performance requirements.

2. COMMAND PROCESSOR CSC

2.1 COMMAND PROCESSOR INTRODUCTION

2.1.1 Command Processor Overview

The Command Processor CSC resides in the CCWS and provides the capability for the CCWS user to issue Command Requests to end items attached to the RTPS. The Command Processor receives CCP Command Responses for each Command Request and displays an appropriate message~~s~~ on the Command Processor screen. The Command Processor ~~and/or~~ issues a message to System Message Services for specific CCP Command Responses.

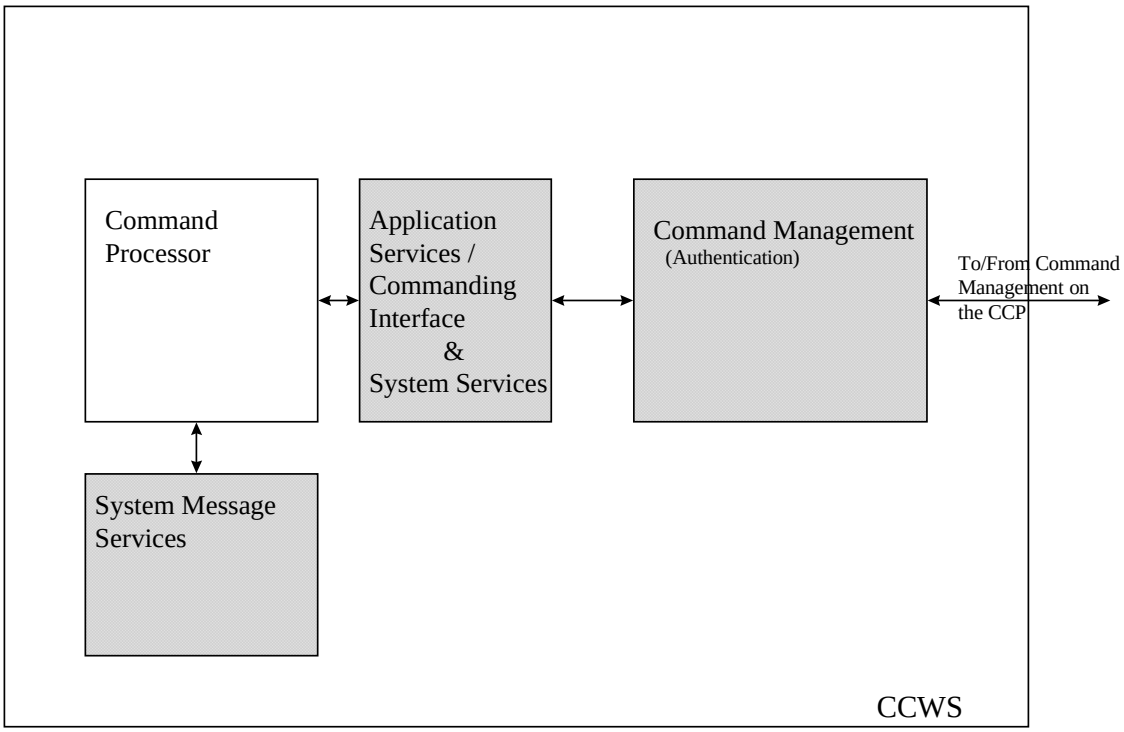


Figure 2~~2~~-1~~1~~ Command Processor Overview

2.1.2 Command Processor Operational Description

The Command Processor provides a means to enter Command Requests to end items attached to the RTPS by a user at ~~an~~ CCWS. ~~The methods by which these Command Requests are entered will be prototype procedures for Redstone.~~

A Command Processor Command Request is made by typing a text string into a field on a CCWS User display screen. The Command Processor parses the text string to create a Command Request. Error messages are displayed indicating syntax or commanding errors. The Command Processor validates whether the Command Request is for an existing FD and is a supported command. A Command Request will be sent if the entry is processed without error and accepted by Two Step Processing.

The Command Processor receives status about Command Requests from several stages in the command route. These messages consist of Two Step Command Processing error/status messages and CCP Command Responses. These messages/responses are displayed on the Command Processor screen and are sent to System Message Services.

2.2 COMMAND PROCESSOR SPECIFICATIONS

2.2.1 Command Processor Ground Rules

2.2.1.1 User Display Services will be required to support the Command Processor to:
- A) provide a modeless pop-up window for 2 step command options.
- B) provide the user the ability to cancel a window.

2.2.1.2 Services will be required to support the Command Processor to provide data to validate command syntax.~~:~~

~~A) search for a FD by FD ID~~
~~A) search for a FD by FD Name~~
~~A) return the following FD attributes:~~
- ~~a) FD ID~~
- ~~a) FD Name~~
- ~~a) Current value~~
- ~~a) FD type~~
- ~~a) Options (for Discrete only - Open/ Close, On/Off, etc)~~
- ~~a) Range (for Analog only - Min, Max, precision)~~
- ~~a) Routing code~~
- ~~a) Destination~~

2.2.1.3 System Services Interprocess Communications will be required to support the Command Processor CSC communications with the Command Management CSC.

2.2.1.4 The Command Processor will output error messages to System Message Services.

2.2.1.5 System Message Services shall provide an API that will return a formatted text string for the Command Processor to display.

2.2.1.6 Command Processor shall be initialized by Control Navigation System.

2.2.2 Command Processor Functional Requirements

2.2.2.1 ~~2.2.2.1 a)~~ The Command Processor shall provide a user interface for command input.
2.2.2.2 The Command Processor interface shall accept keyboard input.
2.2.2.3 Commands shall be case insensitive.
2.2.2.4 A blank delimiter is equal to one or more blank characters.
2.2.2.5 A comma delimiter is equal to exactly one comma.
2.2.2.6 The Command Processor shall issue command requests.

~~2.2.2.2 The commanded objects shall be designated by FD names.~~
~~2.2.2.3 The Command Processor Set command shall use the correct discrete definition for the FD. For example:~~
~~a) Open/Close~~
~~a) True/False~~
~~a) Wet/Dry~~
~~a) On/Off~~
~~2.2.2.4 The Command Processor Apply command shall use numeric input for analog FD's.~~
2.2.2.7 The Command Processor shall accept command syntax for commands marked "Syntax Required" in Table 1 as specified in Appendix A.
2.2.2.8 ~~2.2.2.5~~ The Command Processor shall receive Command Responses from the CCP concerning each Command Request. The responses shall be routed to the User's Screen and System Message Services according the table below.

	CCP Command Response	Command Processor Message	Additional Messages from Command Processor
a	Authentication Problem	Error Message	No Message
b	Command Time-out	Error Message	No Message
c	Command issued	No Message	No Message
d	Command completed	Status Message	No Message
e	Command Error (non-HIM error)	Error Message	Message to System Message Services
f	Command Error (HIM error)	Error Message	No Message

2.2.2.9 The Command Processor shall restrict invalid Command Requests
a) The Command Processor shall~~will~~ reject invalid FD names.
b) The Command Processor shall ~~will~~ reject values out-of-range for an intended FD.
c) The Command Processor shall ~~will~~ reject invalid value types for an intended FD.
2.2.210 The Command Processor shall support two-step Command Requests (arm f~~F~~irst, then execute).

2.2.2.11 The Command Processor shall interface with System Message Services to retrieve command status messages.

2.2.3 Command Processor Performance Requirements

Command Processor Performance Requirements are not defined for the Thor ~~Redstone~~ Release.

2.2.4 Command Processor Interfaces Data Flow Diagram

2.3 COMMAND PROCESSOR DESIGN SPECIFICATION

2.3.1 Command Processor Detailed Data Flow

2.3.1.1 Command Processor Detailed Data Flow

2.3.1.2 Command Processor Class Diagrams

2.3.2 Command Processor External Interfaces

2.3.2.1 Command Processor Message Formats

2.3.2.2 Command Processor Display Formats

2.3.2.3 Command Processor Input Formats

~~The Command Processor shall support the following language formats:~~

~~General rules:~~

~~A. Commands shall be case insensitive. Lower and upper case letters are completely interchangeable. For example: a lower case "a" is equal to a upper case "A".~~
~~A. A blank delimiter is equal to one or more blank characters.~~
~~A. A comma delimiter is equal to exactly one comma.~~

Refer to Appendix A CLCS Commands.

2.3.2.4 Command Processor Recorded Data

N/A

2.3.2.5 Command Processor Printer Formats

N/A

2.3.2.6 Command Processor Interprocess Communications (C-to-C Communications)

2.3.2.7 Command Processor External Interface Calls (e.g., API Calling Formats)

2.3.2.8 Command Processor Table Formats

N/A

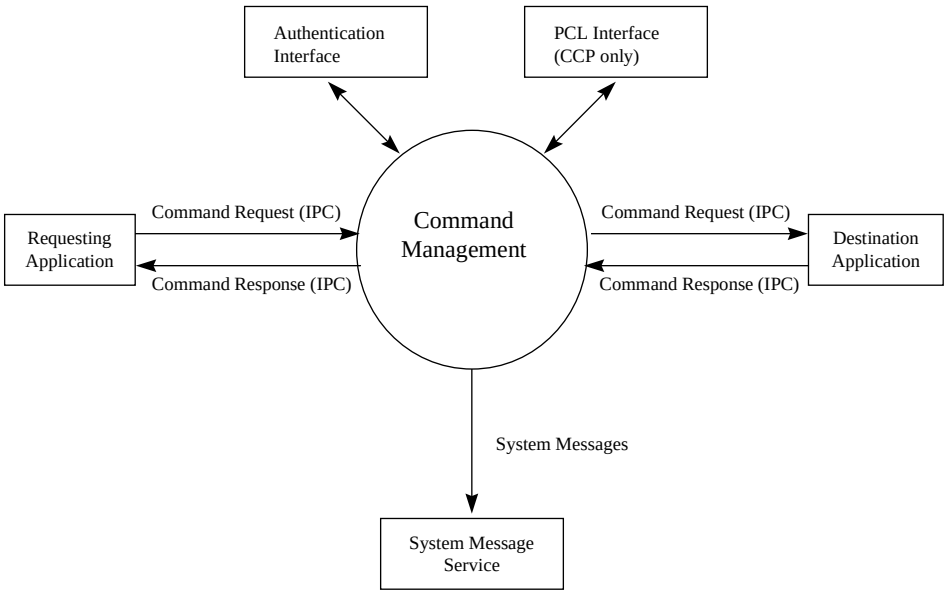
2.3.3 Command Processor Test Plan

3. COMMAND MANAGEMENT CSC

3.1 COMMAND MANAGEMENT INTRODUCTION

3.1.1 Command Management Overview

Command Management is a service that provides Transaction Management of Command Requests and Responses as they are transported through the RTPS System. Commands are received at the Command Management via the Commanding Interface CSC, and are sent back to that interface when the command has been completed. The Command Management process also assures that Command Responses are received from their destination in a timely manner, or a time out message will be sent to the Commanding Interface that initiated the Command Request.
~~Command Management is a service that provides applications the ability to issue Command Requests and receive Command Responses. Command Management monitors the Command Requests for time out conditions allowing applications to perform other tasks.~~



3.1.2 Command Management Operational Description

Command Management has two operation modes depending upon its platform location: CCP or CCWS.

In general, Command Management receives Command Requests, performs ~~some~~ authentication and stores information about the Command Requests into a transaction table. The Command Requests are then forwarded to their destination, and the transaction table is monitored in case the Command Request should time out. Command Responses ~~are~~ received from the destination(s) ~~and~~ are matched up with their entry in the transaction table and ~~to be~~ forwarded back to the requesting process. If a Command Request should time out, a Command Response will be generated with the appropriate error code and a system message will be sent to System Message Services.

Command Management on the CCWS calls the Authentication Interface to verify that the CCWS user application has the authority to issue the command. Command Management on the CCP performs two types of authentication depending on the source of the message. If the message originates from outside the Command Management's CCP, then Test Set authentication is performed. If the message originates from Command Management's CCP, then Command Management calls the Authentication Interface to verify that the CCWS application has the authority to issue the command.

Command Management on the CCP calls Data Distribution APIs to process Pseudo FD commands, change Data Health commands, and change Display Attributes commands.

3.2 COMMAND MANAGEMENT SPECIFICATIONS

3.2.1 Command Management Ground Rules

3.2.1.1 The Command Management will be started and terminated by the System Control CSCI.

3.2.1.2 The Command Management will be initialized on the CCWS and CCP platforms.

3.2.1.3 The Command Management, when initialized, will configure itself to perform certain levels of command authentication depending on which platform it resides on.

3.2.1.4 System Services Interprocess Communications will be required to support the Command Management to:

A) provide communications between system applications residing on the same platform.

B) provide communications between system applications residing on different platforms. ~~Inter-application Communications will make access to different communications services transparent.~~

3.2.1.5 All Command Requests from the Command Management must respond with one of the following Command Responses:

A) Error Response.

B) Successful Response

3.2.1.6 Command Responses to the Command Management must include a Packet Payload Header for each response, formatted in accordance with the Real Time Processing System Packet Payload ICD, Document Number 84K00351. ~~which includes the following data fields:~~

~~A) Payload Type~~

~~A) Flags (Record, Response expected, etc.)~~
~~A) Number of bytes in Response Payload~~
~~A) Place/Flow Number (received in Command)~~
~~A) Logical Source of Sender (received in Command)~~
~~A) Transaction ID (received in Command)~~
~~A) Completion Code~~
~~A) Time~~

3.2.1.7 FD Command Responses to the Command Management must include a Packet Payload Body, formatted in accordance with the Real Time Processing System Packet Payload ICD. ~~which includes the following data fields:~~

~~A) FD ID~~
~~A) FD Commanded Value (represented in Engineering Units.)~~
~~A) FD Value returned from the HIM (represented in Engineering Units.)~~
~~A) FD Commanded Value (represented in Raw Counts)~~
~~A) FD Value returned from the HIM (represented in Raw Counts)~~

3.2.1.8 The Command Management will output error and status messages to System Message Services.

3.2.1.9 The PCL CSC shall provide a PCL API.

3.2.1.10 Data Distribution shall provide an API to update individual FD values.

3.2.1.11 Data Distribution shall provide an API to update a FD's Display Attribute values.

3.2.1.12 Data Health shall provide an API to update a FD's health values.

3.2.2 Command Management Functional Requirements

3.2.2.1 The Command Management shall initialize itself and wait for Command Requests and Command Responses when Command Management is started by the System Control CSCI.

3.2.2.2 The Command Management, on the CCWS platform, shall receive Command Requests from ~~Applications using FD Services~~ the Commanding Interface.

~~3.2.2.3 The Command Management, on the CCWS platform, shall format an CCWS Command Request for each Command Request received.~~

3.2.2.3 The Command Management, on the CCP platform, shall receive CCWS Command Request messages formatted in accordance with the Real Time Processing System Packet Payload ICD.

3.2.2.4 The Command Management, on the CCP platform, shall verify (Authenticate) that the CCWS Command Request is issued from a configured CCWS that is part of the Test Set.

~~3.2.2.5 The Command Management, on the CCP platform, shall format an CCP Command Request for each CCWS Command Request received.~~

~~3.2.2.5 The Command Management shall create a Packet Payload Header, formatted in accordance with the Real Time Processing System Packet Payload ICD, for each Command Request which includes the following data fields:~~

~~A) Payload Type~~
~~A) Flags (Record, Response expected, etc.)~~
~~A) Number of bytes in Request Payload~~

~~A) Place/Flow Number~~
~~A) Logical Source of Sender~~
~~A) Routing ID (Command Type)~~
~~A) Request ID (Command Sub-type)~~
~~A) Transaction Number for Route~~
~~A) Time~~

~~3.2.2.5 The Command Management software shall create a Packet Payload Body, formatted in accordance with the Real Time Processing System Packet Payload ICD, for each Command Request which includes the following data fields:~~

~~A) FD ID~~
~~A) FD value (discrete or analog depending on FD type) (Analog values shall be represented in Engineering Units.)~~

3.2.2.5 The Command Management shall track ~~all~~ Command Requests for response matching for commands that require a Command Response when directed by the RTPS Packet Payload.

3.2.2.6 The Command Management shall keep a TimeOut value for each Command Request sent.

A) The Command Management shall set the TimeOut value to one of the following system default values:

Command Management TimeOut~~Timeout~~ Table	
Location of CM	TimeOut~~Timeout~~
CCWS	120 msec
CCP ~~to GSE G/W~~	50~~10~~ msec

B) The Command Management shall issue an Error Response when a Command Request's TimeOut is exceeded.

3.2.2.11 The Command Management shall accept the following Command Responses:

A) Error Response.
B) Successful Response.
C) Issue Response

3.2.2.12 The Command Management shall accept Unsolicited Command Responses. Unsolicited Command Responses include those responses received for command requests that have timed out.

A) The Command Management shall send a message to System Message Services for each Unsolicited Command Response received.
B) The Command Management shall not attempt to route the Unsolicited Command Response back to the commanding source of the message.

3.2.2.13 The Command Management shall match the Command Responses to the stored Command Requests in the transaction table.

3.2.2.14 The Command Management shall forward Command Responses containing a status or error message to the originating process.

	Response Message (RM)	RM	System

		Returned	Message
a	Invalid Source (Flow) for Command Request	Y	Y
b	GW Command (Error) Response	Y	N*
c	GW Command (Successful) Response	Y	N
d	CMD MGMT Generated Gateway Busy	Y	Y
e	CMD MGMT Generated Command Issued	Y	N
f	CMD MGMT Generated Command Time Out Error Command Response	Y	Y
g	GW Unsolicited Command Responses (error)	Y**	Y

* G/W will issue HIM error messages to System Message Services.
** Issuing source from Packet Header.

3.2.2.15 The Command Management on the CCP shall route commands through the PCL interface.
3.2.2.16 The Command Management shall update the timeout value for a command in the transaction table when processing an "issue response" Command Response.
3.2.2.17 Pseudo FD Translator shall interface with Data Distribution and Data Health APIs.

3.2.3 Command Management Performance Requirements

Command Management performance requirements are defined below:

CLCS System Level Specification paragraph	Requirement.
2.2.2.2.2	RTPS shall be able to support full Uplink command rates on the following links: GSE - 500/second.

The SLS requirement translates to "The Command Management CSC on the CCP will process a command in 16 milliseconds." This requirement is computed as follows:
16 msec = (1 sec) / ((500/second)/(8 CCPs))
where 8 CCPs is the maximum number of CCPs for a Test Configuration.

3.2.4 Command Management Interfaces Data Flow Diagram

3.3 COMMAND MANAGEMENT DESIGN SPECIFICATION

3.3.1 Command Management Detailed Data Flow

3.3.1.1 Command Management Detailed Data Flow

3.3.1.2 Command Management Class Diagrams

3.3.2 Command Management External Interfaces

3.3.2.1 Command Management Message Formats

3.3.2.2 Command Management Display Formats

N/A

3.3.2.3 Command Management Input Formats

N/A

3.3.2.4 Command Management Recorded Data

3.3.2.5 Command Management Printer Formats

N/A

3.3.2.6 Command Management Interprocess Communications

3.3.2.7 Command Management External Interface Calls

3.3.2.8 Command Management Table Formats

N/A

3.3.3 Command Management Test Plan

4. AUTHENTICATION INTERFACE CSC

Thor Command Support provides the initial authentication capabilities. The requirements provided below will be augmented with prototype authentication system support code. As the overall authentication design progresses, the Authentication Interface requirements may be affected.

The purpose of Authentication is to assure that commands are issued only from authorized sources. An authorized source is either an authorized CCWS or an authorized EIM. For example, if a command is issued from a viewer on a CCWS, then the CCWS must be authorized to issue that particular command. If an EIM issues a command, then that EIM must be authorized to issue the command. Notice that the first case associates authority with a physical device and the second case associates authority with an application.

Authentication CSC is invoked by a client application and the resulting disposition is returned to the client. The client must assure that the command does not progress in the system if authority is denied. The client applications are Command Management and the CORBA Filter. Furthermore, a system message will report any authentication disapproval.

There are several types of commands which must be authenticated. They include FD commands, commands associated with FDs, commands associated with Gateways, and commands invoked via an ORB (Object Request Broker). Furthermore, there may be commands which do not fit one of these categories.

4.1 AUTHENTICATION INTERFACE INTRODUCTION

4.1.1 Authentication Interface Overview

The interface to Authentication is bi-directional. The input shall consist of the command and the necessary information to authenticate the command. For the purposes of this discussion, the input information is collectively referred to as the command information. The output consist of a disposition of the authentication request. Additionally, a system message is sent for negative authentication responses.

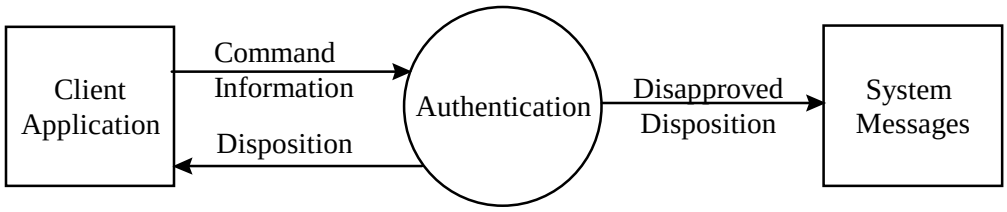


Figure 4444-1111 : Authentication Interface Data Flow

4.1.2 Authentication Interface Description

4.2 AUTHENTICATION INTERFACE SPECIFICATIONS

4.2.1 Authentication Interface Ground Rules

The ground rules define the basis for developing the Authentication Interface. The specific rules follow:

4.2.1.1 For CCWS invocations of Authentication Interface, the sub-system identifier shall be passed via the interface.
4.2.1.2 For EIM invocations of Authentication Interface, the application name shall be passed via the interface.
4.2.1.3 The client of the Authentication interface must properly handle negative responses. It is very important to note that it is not Authentication's responsibility to stop the processing of a command. Therefore, the client must stop the progression of the command based on a negative authentication response.

4.2.2 Authentication Interface Functional Requirements

The interface input/output follows:

4.2.2.1 The input to the Authentication Interface shall be the command information. The command information shall not be modified by the Authentication Interface.
4.2.2.2 The output of the Authentication Interface shall consist of one of the following return responses:
A) Command is authorized.
B) Command is not a member of the TCID.
C) Command is not associated with a user class assigned to the CCWS.
D) Command is not authorized for the application.
E) Erroneous command information.
4.2.2.3 If the command is not authorized, then the Authentication Interface shall issue a system message with the following information:
A) Reason for the disapproval:
1) Command is not a member of the TCID.
2) Command is not associated with a user class assigned to the CCWS.
3) Command is not authorized for the application.
4) Erroneous command information.
B) The name of the command.
C) For CCWS initiated commands, the name of the initiating sub-system.
D) For EIM initiated commands, the name of the initiating application.
4.2.2.3 The interface to the Authentication Interface shall support invocation on CCWSs and CCPs.

4.2.3 Authentication Interface Performance Requirements

Authentication Interface performance requirements are not defined. The Authentication Interface shall process an authentication request in a period of time that does not impact the requirements specified in section 3.2.3.

4.2.4 Authentication Interface Interfaces Data Flow Diagram

Figure 4-1 shows the data flow for the interface to the Authentication Interface. The command information is passed to authentication and the disposition is returned. A system message is only sent if a disapproval occurs.

4.3 AUTHENTICATION INTERFACE DESIGN SPECIFICATION

4.3.1 Authentication Interface Detailed Data Flow

4.3.1.1 Authentication Interface Class Diagrams

4.3.2 Authentication Interface External Interfaces

4.3.2.1 Authentication Interface Message Formats

4.3.2.2 Authentication Interface Display Formats

N/A

4.3.2.3 Authentication Interface Input Formats

N/A

4.3.2.4 Authentication Interface Recorded Data

4.3.2.5 Authentication Interface Printer Formats

N/A

4.3.2.6 Authentication Interface Interprocess Communications

4.3.2.7 Authentication Interface External Interface Calls

4.3.2.8 Authentication Interface Table Formats

N/A

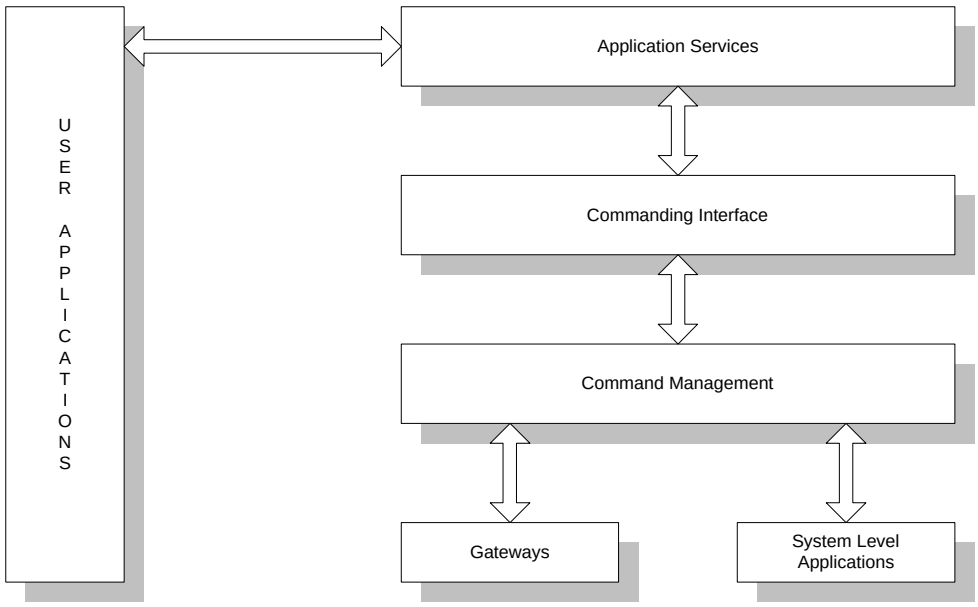
4.3.3 Authentication Interface Test Plan

5. COMMANDING INTERFACE CSC

5.1 COMMANDING INTERFACE INTRODUCTION

5.1.1 Commanding Interface Overview

The Commanding Interface CSC consists of a set of System Level APIs used for sending commands through the RTPS System. The Commanding Interface APIs send Command Requests and Responses to a Command Management process which manages the status of the commands. Responses to the Command Requests are returned to the requesting applications when completed.



Command Interface Overview

5.1.2 Commanding Interface Description

The purpose of the Commanding Interface CSC is to provide a means of sending commands from User Applications to End Items or System Level Applications in the RTPS System, without needing specific knowledge of the underlying Systems Services. Command Requests and Responses are transported to a Command Management Process which monitors all transactions the commands encounter to and from their destinations. The Command Responses are returned from the Command Management Process with a completion code that represents the success or failure of the Command Request.

The Commanding Interface APIs are provided in an Object Oriented fashion so that commands are “logically grouped” according to the purpose of the command. For instance,

commands that control specific FDs are provided in a FD Commanding Object, commands that control table maintenance on a Gateway are provided in a Gateway Commanding Object, and so on. The Commanding Interface APIs are accessed through the Application Services layer, which contain a set of APIs for User Applications.

5.2 COMMANDING INTERFACE SPECIFICATIONS

5.2.1 Commanding Interface Ground Rules

5.2.1.1 Commands will be verified by the application for content/grammar prior to calling the Commanding Interface APIs.

5.2.1.2 OLDB shall provide initial routing information to a CCP and Gateway for all FD's, including Pseudo FD's.

5.2.1.3 Access to the Commanding Interface APIs will be provided through the Application Services CSCI Interface.

5.2.1.4 The RTPS Packet shall provide a field to specify routing of LDB Commands to a priority queue on the LDB.

5.2.2 Commanding Interface Functional Requirements

5.2.2.1 The Commanding Interface shall provide a means to specify PCL Override on FD Commands.

5.2.2.2 The Commanding Interface shall support Ground Support Equipment Commands.

- 5.2.2.2.1 **APPLY (Apply Analog)**
- 5.2.2.2.2 **SET (Set Discrete)**
- 5.2.2.2.3 **ISSUE (Issue Digital Pattern)**
- 5.2.2.2.4 **R FD (Read GSE HIM Output FD)**
- 5.2.2.2.5 **R EUC (Read EU Coefficients)**
- 5.2.2.2.6 **A/I HT (Activate/Inhibit HIM Testing Command)**
- 5.2.2.2.7 **A DA (Activate Data Acquisition)**
- 5.2.2.2.8 **I DA (Inhibit Data Acquisition)**
- 5.2.2.2.9 **A/I PR (Activate/Inhibit Data Processing)**
- 5.2.2.2.10 **A/I CMD (Activate/Inhibit Command Issuance)**
- 5.2.2.2.11 **A/I HI (Activate/Inhibit HIM Polling Command)**
- 5.2.2.2.12 **A/I SI (Activate/Inhibit Significant Change Check)**
- 5.2.2.2.13 **C HA (Change Hardware Address)**
- 5.2.2.2.14 **C RA (Change Sample Rate)**
- 5.2.2.2.15 **C EUC (Change EU Coefficients)**
- 5.2.2.2.16 **S FD (Status FD)**

5.2.2.3 The Commanding Interface shall provide Initial Launch Data Bus Commands.

- 5.2.2.3.1 **APPLY (Apply Analog)**
- 5.2.2.3.2 **SET (Set Discrete)**
- 5.2.2.3.3 **ISSUE (Issue Digital Pattern)**
- 5.2.2.3.4 **CBTU MR (LDB MDM Master Reset)**
- 5.2.2.3.5 **CBTU LBSR (LDB Load BITE Status Register)**
- 5.2.2.3.6 **CBTU BT1 (LDB BITE Test 1)**
- 5.2.2.3.7 **CBTU BT3 (LDB BITE Test 3)**

5.2.2.3.8 **READ (Read Onboard Values)**
5.2.2.3.9 **CBTU RBSR (LDB Read BITE Status Register)**
5.2.2.3.10 **CBTU WRAP (LDB Return Received Command Word)**
5.2.2.3.11 **CBTU BT2 (LDB BITE Test 2)**
5.2.2.3.12 **CBTU BT4 (LDB BITE Test 4)**
5.2.2.3.13 **A DA (Activate Data Acquisition)**
5.2.2.3.14 **I DA (Inhibit Data Acquisition)**
5.2.2.3.15 **EQ DEU ************
5.2.2.3.16 **GPCC LDBC (LDB Control)**
5.2.2.3.17 **TEXT (LDB Send DEU Text Message to GPC)**
5.2.2.3.18 **GMEM** Read *****
5.2.2.3.19 **GMEM** Write *****
5.2.2.3.20 **S FD (Status FD)**

5.2.2.4 The Commanding Interface shall provide a means of routing commands to a Priority Queue for LDB Commands.

5.2.2.5 The Commanding Interface shall provide Initial PCM Commands.

5.2.2.5.1 **R EUC (Read EU Coefficients)**
5.2.2.5.2 **A DA (Activate Data Acquisition)**
5.2.2.5.3 **A/I PR (Activate/Inhibit Data Processing)**
5.2.2.5.4 **A/I SI (Activate/Inhibit Significant Change Check)**
5.2.2.5.5 **A FL (Activate Frame Logging)**
5.2.2.5.6 **I FL (Inhibit Frame Logging)**
5.2.2.5.7 **C EUC (Change EU Coefficients)**
5.2.2.5.8 **C PSB (Change Sync Bits in Error Count)**
5.2.2.5.9 **PCMS (Change/Select PCM)**
5.2.2.5.10 **S FD (Status FD)**

5.2.2.6 The Commanding Interface shall provide initial commanding to support End Item Manager Control.

5.2.2.7 The Commanding Interface shall format Command Requests in accordance to the RTPS Packet Payload ICD, Document Number 84K00351.

5.2.2.8 The Commanding Interface shall accept Command Responses formatted in accordance to the RTPS Packet Payload ICD, Document Number 84K00351.

5.2.3 Commanding Interface Performance Requirements

N/A

5.2.4 Commanding Interface Interfaces Data Flow Diagram

N/A

5.3 COMMANDING INTERFACE DESIGN SPECIFICATION

5.3.1 Commanding Interface Detailed Data Flow

N/A

5.3.1.1 Commanding Interface Class Diagrams

5.3.2 Commanding Interface External Interfaces

5.3.2.1 Commanding Interface Message Formats

5.3.2.2 Commanding Interface Display Formats

N/A

5.3.2.3 Commanding Interface Input Formats

N/A

5.3.2.4 Commanding Interface Recorded Data

5.3.2.5 Commanding Interface Printer Formats

N/A

5.3.2.6 Commanding Interface Interprocess Communications

5.3.2.7 Commanding Interface External Interface Calls

5.3.2.8 Commanding Interface Table Formats

N/A

5.3.3 Commanding Interface Test Plan

6. COMMAND SCRIPTER CSC

6.1 COMMAND SCRIPTER INTRODUCTION

6.1.1 Command Scripter Overview

The Command Scripter is a GUI application that will allow users to Enter, Edit and Execute text files containing simulated operator inputs. These files are used to store commands in a CPRO format, just as they would be entered from the command line.

6.1.2 Command Scripter Description

The function of the Command Scripter is to read simulated operator inputs from a disk file and feed them into the CLCS System as if the operator were keying them in manually. A GUI interface will be provided to view the script files, and allow the operator to edit and add lines to the file as needed. The GUI interface will also allow the operator to execute the file using Control Buttons which can control the execution of the file, such as changing mode, changing time delays, and setting breakpoints.

The Command Scripter will also accept imbedded commands from the disk files. This special set of commands can control the execution of the file, much like the Control Buttons on the GUI interface.

6.2 COMMAND SCRIPTER SPECIFICATIONS

6.2.1 Command Scripter Ground Rules

6.2.1.1 Command Scripts will be stored as ASCII text files.
6.2.1.2 Commands supported by the Commanding Scripter CSC will consist of, not to exceed, commands listed in Appendix A.
6.2.1.3 Mode Control Commands entered in the script files will be surrounded by the ‘<’ and ‘>’ characters. i.e. <DELAY 10>.

6.2.2 Command Scripter Functional Requirements

6.2.2.1 The Command Scripter CSC shall provide a GUI Interface for editing/executing script files.
6.2.2.2 The Command Scripter shall provide syntax checking on the script files.
6.2.2.3 The Command Scripter shall display any errors in the syntax when the file is loaded onto the GUI Interface.
6.2.2.4 The Command Scripter shall provide a manual input process for editing or entering commands.
6.2.2.5 The Command Scripter shall accept ASCII script files created from another source.

6.2.2.6 The Command Scripter shall provide the ability to Save a new or modified script file.

6.2.2.7 The Command Scripter shall provide an "Override" capability from the GUI Interface to change the mode of execution of the script file.

6.2.2.8 The Command Scripter shall accept a set of Mode Control Commands which can be embedded in the script file. These Mode Control Commands are:

6.2.2.8.1 <DELAY X> or <**DEL X>** where X is the number of milliseconds to delay between executable lines.

6.2.2.8.2 <FAST> or <**FAS>** to change the mode of execution, executing as fast as the system can handle.

6.2.2.8.3 <SLOW> or <**SLO>** to change the mode of execution, pausing for 3 seconds between executable lines.

6.2.2.8.4 <SINGLE STEP> or <**SS>** to change the mode of execution, executing one line at a time, pausing for operator action between every executable line.

6.2.2.9 The Command Scripter GUI Interface shall have a set of mode control buttons for controlling the execution of the script file. These Mode Control Buttons are:

6.2.2.9.1 START Button - starts the execution of the script file.

6.2.2.9.2 STOP Button - temporarily stops execution of the script file.

6.2.2.9.3 CONTINUE Button - when a script file has been stopped during execution, or in single step mode, will proceed to execute the next executable line if one exists.

6.2.2.9.4 FAST Button - changes the current mode of execution to fast.

6.2.2.9.5 SLOW Button - changes the current mode of execution to slow.

6.2.2.9.6 SINGLE STEP Button - changes the current mode of execution to single step.

6.2.2.9.7 SET BREAK POINT Button - toggles a break point for a given executable line which will temporarily stop execution at that line when the script file is run.

6.2.2.10 The Command Scripter shall temporarily stop execution of a script file upon error in command execution and wait for action from operator.

6.2.3 Command Scripter Performance Requirements

N/A

6.2.4 Command Scripter Interfaces Data Flow Diagram

N/A

6.3 COMMAND SCRIPTER DESIGN SPECIFICATION

6.3.1 Command Scripter Class Diagrams

6.3.2 Command Scripter External Interfaces

6.3.3 Command Scripter Message Formats

6.3.4 Command Scripter Display Formats

6.3.5 Command Scripter Input Formats

6.3.6 Command Scripter Recorded Data

6.3.7 Command Scripter Printer Formats

6.3.8 Command Scripter Interprocess Communications

6.3.9 Command Scripter External Interface Calls

6.3.10 Command Scripter Table Formats

6.3.11 Command Scripter Test Plan